Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A method in a computer ~~data processing~~ system for modifying ~~processing~~ instructions using [[by]] a processing unit that has a standard instruction set, the method comprising:

using, by an encryption algorithm each time the computer ~~data processing~~ system is rebooted, a different one of a plurality of different instruction maps to dynamically remap the standard instruction set to create a new instruction set; [[and]]

programming a decode unit to decode code using a selected one of the plurality of different instruction maps that was selected in response to a particular reboot of the computer system, wherein a particular new instruction set was created using the selected one of the plurality of different instruction maps;

receiving a selection of particular code to be executed by a processor that includes the decode unit;

determining whether or not the particular code is trusted code, wherein code is trusted only when the code resides in an area of the computer system that is trusted to be free of malicious code, and further wherein the area includes a program loader and the plurality of different instruction maps;

in response to determining that the particular code is trusted code: using, by a remapping process in the program loader, the selected one of the plurality of different instruction maps to remap each opcode in the trusted code to new opcode using the particular new instruction set to produce encoded code, wherein the decode unit can decode the encoded code;

in response to determining that the particular code is not trusted code, leaving each opcode in the particular code unchanged, wherein the decode unit cannot decode the particular code; and

processing, by the processor that is included in the processing unit, only those instructions that use the particular new instruction set.

2. (Currently amended) The method of claim 1, further comprising:

performing the dynamic remapping during execution of an initial program load (IPL) process and before the computer ~~data processing~~ system begins executing an operating system.

3. (Previously Presented) The method of claim 1, wherein each one of the plurality of different instruction maps is an opcode map.

4. (Previously Presented) The method of claim 1 further comprising:

encoding a set of instructions from a trusted computer base using the one of the plurality of different instruction maps to form a set of encoded instructions; and

sending the set of encoded instructions to the processing unit for execution.

5. (Canceled)

6. (Original) The method of claim 4, wherein the encoding step and the sending step are performed by a program loader.

7. (Previously Presented) The method of claim 1 further comprising:

responsive to an event, executing a process to select the one of the plurality of different instruction maps.

8. (Previously Presented) The method of claim 7, wherein the process uses a machine serial number and a number of boot cycles to select the one of the plurality of different instruction maps .

9. (Currently amended) The method of claim 7, wherein the event is at least one of an initialization of the ~~data processing~~ computer system and a user input.

10. (Previously Presented) The method of claim 1, wherein the new instruction set is created using a first one of the plurality of different instruction maps when code is executed by a first privilege level and wherein a second one of the plurality of different instruction maps is used when code is executed by a second privilege level.

11-25. (Canceled)

26. (New) A method in a computer system for modifying instructions using a processing unit that has a standard instruction set, the method comprising:

using, by an encryption algorithm each time the computer system is rebooted, a different one of a plurality of different instruction maps to dynamically remap the standard instruction set to create a new instruction set;

determining whether or not particular code is trusted code, wherein code is trusted only when the code resides in an area of the computer system that is trusted to be free of malicious code, and further wherein the area includes a program loader and the plurality of different instruction maps;

in response to determining that the particular code is trusted code: using, by a remapping process in the program loader, the selected one of the plurality of different instruction maps to remap each opcode in the trusted code to new opcode using the particular new instruction set to produce encoded code; and

in response to determining that the particular code is not trusted code, leaving each opcode in the particular code unchanged.